A big data analytics based approach to anomaly detection

Razaq, Abdul; Tianfield, Huaglory; Barrie, Peter

Published in:
Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies

DOI:
10.1145/3006299.3006317

Publication date:
2016

Document Version
Author accepted manuscript

A Big Data Analytics Based Approach to Anomaly Detection

Abdul Razaq
Department of Computer, Communications and Interactive Systems, Glasgow Caledonian University, UK
abdul.razaq@gcu.ac.uk

Huaglory Tianfield
Department of Computer, Communications and Interactive Systems, Glasgow Caledonian University, UK
h.tianfield@gcu.ac.uk

Peter Barrie
Department of Computer, Communications and Interactive Systems, Glasgow Caledonian University, UK
peter.barrie@gcu.ac.uk

ABSTRACT
We present a novel Cyber Security analytics framework. We demonstrate a comprehensive cyber security monitoring system to construct cyber security correlated events with feature selection to anticipate behaviour based on various sensors.

Keywords
Event Correlation, Process Auditing, IDS/IPS, SIEM, Advanced Persistent Threats, Security Analytics.

1. INTRODUCTION
Many of the existing security measurements are insufficiently scalable, incompatible, or simply inadequate to address the challenges that are posed in highly complex environments. On the one hand, security of systems and subsystems is important but it should not result in exhausted unreliable or underperforming system. On the other hand, efficiency and reliability should not be achieved at the cost of vulnerabilities.

The remainder of the paper is arranged as follows. Literature review is presented in section 2 to evaluate existing security practices. In section 3, we present our system design followed by system architecture and discussion on framework itself before summarizing our work with applications and initial results. A detailed discussion is presented in section 5 to conclude this paper.

2. LITERATURE REVIEW

2.1 Anomaly Detection
Intrusion Detection Systems (IDS) are set of the procedures to monitor the events occurring in a computer or network and examine these events for signs of unwanted activities. These unwanted activities can be categorized such as illegitimate surveillance, un-authorized access to remote or local resources and denial of service. Detection methods fall into two categories: Signature - a known malicious activity, and Anomaly - a deviation from regular patterns.

Signature based systems are conventional solutions against known threats, generally referred to blacklisted intruders. For example, a firewall will keep list of IPs that would not be entertained and shall be blocked. This is a similar method implemented in most of the antiviruses or antimalware solutions to detect unwanted system events. Signature database can consist of rules including IP, MAC addresses, port numbers, user IDs or list of Common Vulnerabilities and Exposures (CVE) etc.

Anomaly detection requires software architecture with cognitive algorithms. Typical behaviour based solution will monitor user activity, network traffic or native system activity to compare against any deviation from established model. The models are generally split into two broad categories; legitimate and abnormal. Intrusion is considered when there is a deviation from legitimate pattern trending into abnormal boundaries. This is analogous to a standard deviation from mean values. These (Intrusion Detection and Prevention System) IDPSs are far more effective compare to signature based solutions to cope threats such as DDoS, zero-day exploits, however, inaccuracy is huge problem with these solutions.

Patel et al. [1], have presented a systematic review of existing IDPS techniques and how they are insufficient for Cloud platform. Systematic analysis covers basic layers of Cloud computing with detailed requirements that each layer is vulnerable. Classic IDPS systems are unable to address security challenges that are presented in mix-network-topology, multi-user and mixture of software layers in Cloud platforms. Mohamed et al. [2] have adopted to monitor system calls generated by virtual machine's programs to the hypervisor. The proposed approach monitors well-known programs with typical system call sequence, an anomaly is detected if there are any extra system calls or change in sequence. It is a sufficient solution where number and type of application are well-known but false alarm would be a huge problem with non-documented applications. This technique also suffers with typical system overloading and lower throughput problem because of monitoring all the systems calls and their call sequence. Prajapati et al. [3] have critically investigated classical solutions and concluded that databases of incident used to detect anomaly have low reliability. Because, the trainer part of system needs to be updated before it can detect the new attack which again needs to be populated into database. Suggested approach will drastically increase the database size and can introduce latency in monitor and detector functionality.

Kabir et al. [4] proposed statistical framework of IDS based on Fuzzy Network algorithm. First, required dataset size based on selected characteristics is selected from whole dataset, and then selected dataset is further divided into predetermined subgroups to samples from these clusters using the derived optimum allocation scheme. Finally, these samples are used as input to LS-SVM IDS to detect different attacks. This solution was evaluated on sub-set of data which could lead to different results when evaluated against whole data set.

Thaseen et al. [5] proposed integrating Principal Component Analysis (PCA) and Support Vector Machine (SVM) by optimizing the Radial Basis Function (RBF) parameters using automatic parameter selection technique. It is being suggested that minimum resources are consumed as the classifier input requires reduced feature set and thereby minimizing training and testing overhead time. Only two types of attacks were investigated, the outcome could be with different results when applied to complete dataset. Shang-Fu et al. [6] investigated recursive SVM to emulate IDS with high accuracies

compared to SVM without recursion. However, authors have pointed that it is not an ideal solution with distinguished irregularities. Xue-qin et al. [7] experimented with Fisher Score to reduce dimensions of the feature space. However, authors consider that training time for such solution is high. Lonea et al. [8] [9] proposed a multilevel technique with Dempster–Shafer theory (DST), which offers an alternative possibility to anticipate the probability of events based on frequency of events. This technique is useful when similar information is provided by multiple sources which can reduce false alarms. Thus, it reduces the number of detections while combining the similar threats. This approach may fail if trust system is compromised, e.g., similar reported attack with different weight will be undetected.

2.2 Anomaly Detection with Machine Learning

Mehmood et al. [10] summarized anomaly based systems that can be implemented via various machine learning techniques such as: 1) Fuzzy Logic (FL) - uses true or false to detect anomaly, 2) Artificial Neural Networks (ANN) - weighs various inputs and transform them until required output is achieved, 3) Support Vector Machine (SVM) – classifies normalized data via appropriate kernels to divide data into two categories resulting anticipation of new data inputs, 4) Association Rule (AR) - finds a correlation between different data sets, 5) Genetic Algorithm (GA) - searches heuristically to build mutation and crossover genomes based on existing or new genes, 6) K-Means - classifies data into clusters that present average of data based on provided means. These are few widely used practices but the key in all machine learning methods is an ability classify data with certain patterns defined by rules.

Li et al. [11] suggested neural settings for distributed IDS deployed on Cloud. Overhead of anomaly detection would require huge resources which could be function of available resource. Maiti et al. [12] presented simple but practical implementation of Cloud based IDS. Intrusion is detected with image processing via digital camera which fires an alarm if there is a change in frame. This technique is widely used in physical security domain and has its own limitations such as intensive false alarms because image processor will report an abnormality whenever there is a change in frame form pervious frame. However, the presented idea provides a simple yet robust example how cloud based security service can leverage on vast resources for resource limited entities. Tupakula et al. [13] elaborated design suggestions for virtual machine based IDPS targeted for IaaS. The proposed technique captures updates of operating systems and applications on virtual machines to detect unusual activity. Proposed design Supervised learning includes a training data to create a benchmark that is further used for new data. For example, a calculated mean of data can be used to identify if new data point is above or below computed mean. Typical supervised systems will update itself with all new data points; thus training dataset is generally whole dataset. However, some systems do not have to update trainer if scope of data is definite. On the other hand, unsupervised method requires concrete function to classify the data.

Patel et al. [1] systematically reviewed intrusion detection and prevention in three layers of cloud computing service architecture, these included services inter-dependent infrastructure, platforms and applications. Authors presented a conceptual framework based on automatic computing, ontology, risk management and fuzzy theory. Xiong et al. [19] proposed a theoretical system to detect network wide anomaly with combination of synergetic neural networks and the catastrophe theory. Synergetic technique describes complex behaviours of networks and catastrophe method describes dynamic process of the network in cloud communication.

Example of such a system could be a spam filter that only allows emails form trusted list.

The dataset which includes normal and anomalous data points are used to build a predictive model to categories normally with two definite boundaries: required or undesired. Strict supervised method has more overhead compared to unsupervised or semi-supervised techniques. Generally, accuracy is better in supervised methods at the cost of processing and store. Semi-supervised or hybrid supervised system can build a function from historic data to anticipate class of future data, whereas, unsupervised should not require any historic information and would be able to organize data with predefined function. For example, all IP address from source ‘A’ should not be entertained. Unsupervised technique is suitable procedure where historic data is not available and it is also best solution with limited processing and store, therefore it is ideal system for resource limited systems without learning/training overhead. The problem with unsupervised is the inability to predict and accommodate unexpected event as function to define boundaries of legitimate and pretentious are predefined at compile time [14]. Pavel Laskov et al. [15] conducted a study to review supervised and unsupervised methods in intrusion detection techniques.

2.3 Data Normalization and Feature Selection

Data normalization is pre-process which plays a key a role to improve efficiency of employed machine learning algorithm. Normalization is broad technique which could include, sort functions, lower or higher boundaries, data mining techniques, compression, correlation etc. The key idea is to prepare raw observations in a format that would allow to input these values to algorithm. The important aspect is to prepare raw data in readily input which can be directly processed by algorithm without any unnecessary data handling. These data normalization techniques are referred as ‘kernels’ in data mining world that play very important but passive role to marginalize raw observations into meaningful inputs – despite passive role the selection of data to be processed – by machine learning to efficiently process.

Blum et al. [16] presented a review on feature selection methodologies which are crucial for any ML algorithms. They have proposed general framework to compare different methods to evaluate ML algorithm based on relevant features and scenarios. The selection of features and how these should be combined is important and can significantly affect the system throughput. Wen et al. [17] systematically analysed ML models with respect to software development effort estimation (SDEE) from period of 1991–2010 and concluded that ANN and DT are most frequently used algorithms.

Puri et al. [18] investigated graph techniques to detect anomalies with network security log traces provided by Security Information and Event Management (SIEM) vendors. The testbed was setup with Hadoop and python toolsets, and visualization of output was rendered by java-script D3.js framework. It has been suggested that some discovered anomalies were unexpected to SIEM data, and the deficiency has been left to further work.

Jeun et al. [20] presented an informational study on Advanced Persistent Threat (APT) including analysed attack types and common patterns of the APT incidents based on information, countermeasures for the APT. Li et al. [11] proposed anomaly based solution with neural settings to artificially sense the acting. Neural configuration helps to distribute the work load of IDS to spread algorithm across Cloud rather chocking a single machine. It has been claimed that new threats are detected by proposed system with fair accuracy. Overhead of anomaly detection would require huge resources which could be function of available resource. It can be said, that detection

system based on pattern analysis is direct function of provided resources. The more detection hits in the systems would mean occupied resource in given setting. This technique of load balancing is natural as far as all entities share the same interest and will participate with resources and time.

2.4 Event Correlation

The event correlation formed from different sensors is a practice to correlate and link similar characteristic that are related. The linkage diminishes false alarms from localized data-set with elimination of redundant streams of data. This unique structure enables to discover state of regression in monitored system with historic evidence based on machine learning analysis. The ability to construct correlated activities from a vast set of data pool ensures pragmatic approach to identify the unusual activity without any known trail. Inclusion of as many sensors as possible [21] enables to form a complex alert mechanics based on various quantities and qualities.

The work conducted by Wu et la. [22] integrates a number of component techniques to collect time-series data. A time-series data can be defined as a continuous stream of data over time. A fusion algorithm is being deployed for analytics to identify an event constructed with matrix theory. Jiang et al. [23] introduced process query system based on high level observations. Their approach relies on higher-level sensors which have an event constructed such as key press, mouse click, soft or hard interrupt routines. A similar approach was adopted by Irfan et al. [24] which relies on processed data where information is already in a format and correlation has been constructed by collector. The processed information from high-level sensors (e.g. IDPS) certainly makes pragmatic argument but it ignores basic principles of security: “a compromised information can only lead to flawed conclusion”.

3. PROPOSED APPROACH

Big Data analytics based Cyber Security is a Cyber Security Analytics (CSA) architecture based on Network Log (NetL) and in-memory Process Log (PrcL). The primary aim of cyber security analytics architecture is to identify an anomaly vector with assistance of comprehensive system observations. Fig. 1 presents the architecture of CSA with detailed discussion of individual component in rest of this section.

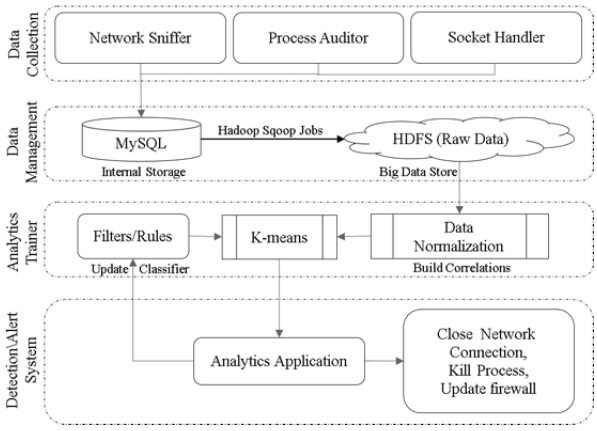

Figure 1. Cyber Security Analytics (CSA) Architecture

A. Feature Selection and Data Collection

CSA gathers enormous variety of information with versatility and with different velocities from NetL and PrcL. The sensors produce unique functionality with exclusive role – if acquired and correlated - can predict system behaviour; nevertheless, pointing to definite activity in past. This continuous monitoring produces compound set of observations – holistic in nature - with infinite amount and inconstant velocities. Thus, important requirement of CSA is to deal with big data in terms of storing and processing.

The feature selection in CSA can be visualized as structure of tree to understand correlations that are being assembled with various sensors to formulate an event. A shared Correlation *(sc)* is something that has resemblance of roots. The metamorphosis of root to trunk represents first degree Variant Correlation *(vc1)*. The second degree variant *(vc2)* is from the same tree with same bark colour and texture; however, branches are above ground and can unnecessarily grow in free space. Finally, and most peculiar is; a third degree variant *(vc3)* which is essentially part of same tree – without any substantive commonality – as they share same roots, trunk and branches. Thus, they can still be classified in unique category as leave and stem would not share any common feature, besides being physical part of tree. The selection of features can be distributed into further discrete diversified categories with the cost of storage and CPU time. The introduced collection mechanism serves as a proof-of-concept to provide a wider outlook of efficiency and uniqueness of such a system.

Network Log (NetL) records, all raw network data in form of packets travelling to and departing from system. The feature organizations aid to fabricate normalization – finding correlation with search and match - to identify similarities. The NetL sniffs data without actively regulating, netfilters or kernel module is recommended approach to override access to target resources. However, the time to execute basic file I/O or common database commits on network packet within kernel space can reduce the performance drastically. Nevertheless, whatever approach is adopted - to capture in or out packet – all are stored in local store.

The analytic assembled from NetL holds unique records from all stored records with correlation in four distinct events: Shared Correlation *(sc)*, Variant Correlation with first degree up-to third labelled as *(vc1, vc2 and vc3)*. The test data was generated with random web access and Apache *‘ab’* benchmarking device.

```
e = Ethernet = {eth_dhost , eth_shost, eth_type}
i = IP = {ip_dst , ip_src, ip_tos, ip_p}
a = Application = {tcp_dport, tcp_sport, tcp_flags}

      if(e & i & a) sc++
      if(e & i & !a) vc1++
      if(e & a & !i) vc2++
      If(i & a & !e) vc3++
```

Figure 2. NetL Feature Selection and Correlation Build-up

We attempt to construct *‘4’* – proof of concept - correlation from *‘3’* unique sets of network packets, as presented in Fig. 2. The *ethernet* set denoted by ‘e’ consists on physical address of source and destination with type of physical media. *IP* set includes destination and source IP addresses, type of service and protocol type. The third last set represents *Application,* with TCP type traffic with destination and source port along flag type.

The Process Log (PrcL) is a component to audit system processes in instant memory, proposed module implements snapshot mechanism with constant interval to capture all the process running in the system. The PrcL prototype is implemented in a bash-script that consists on *‘ps’* command for *‘uname, uid, pid, ppid, session, stime, tty, cmd and tpgid’* columns. This is proof-of-concept and an actual system should

monitor *'/proc'* directory for reduced latency and avoid injection threat. A detailed threat should over-ride process reporting mechanism with infected *'ps'*.

```
if (USER & UID & PID & PPID & SESS & TT & CMD  ) sc++
if (USER & UID & !PID & PPID & SESS & TT & CMD  ) vc1++
if (USER & UID & !PID & PPID & SESS & TT & !CMD  ) vc2++
if (USER & PPID & SESS) vc3++
```

Figure 3. PrcL Feature Selection and Correlation Build-up

The unique sets as illustrated in Fig. 3 are constructed from process list, 1) Existing process in system is considered as a shared correlation *'sc'* if it has a same username, user ID, process ID, parent process ID, session, terminal type and command line arguments, 2) the first degree variant correlation *'vc1'* is similar to *'sc'* with exception to process ID, this employs that user with previous credentials and conditions initiated a new process, 3) Second degree variant correlation *'vc2'* presents a condition where user is initiating a new process with new command-line arguments. And finally, 4) weakest correlation is constructed *'vc3'* if same user from existing session executes a process with same parent process ID.

B. Big Data Store and Management

The storage and processing in CSA is based on distributed framework, as datasets from various sensors (e.g. NetL, PrcL) produce a set of correlated information which falls into its scope. An individual source of log provides unique but correlated observations, for example NetL sniffs packets crossing network boundary or PrcL records all the current running processes in the system. The data from individual sensor is captured in raw format with very high frequency and further channelled to analyser of relevant component. The local store of a device is used as a flip-flop buffer, before it is uploaded to HDFS, the interval to capture data should be limited to avoid short-lived processes or instant packets crossing network boundaries.

All these different types of observations are produced with different velocity, even a single data type can have varied velocity, for example - number of packets will increase if monitored system is being under intense usage, similarly the output from all sensors will vary depending upon their usage. The selected tool-set is not only sufficient but also favourable to reduce processing with built-in facility of MapReduce provided by HDFS. The HDFS is a virtual abstraction of underlying file system that is visible from hadoop echo-system.

In CSA, data is temporarily stored locally in flip-flop buffers with user defined limit which is configurable depending upon case by case. Once the limit is reached, secondary store shall be made available for further commits and primary store will be set to off-load to HDFS via Sqoop job. Sqoop allows transferring data between Hadoop and structural and relational stores.

C. Analytics and Trainer – Machine Learning

The purpose of analyser is to find possible correlations from recorded events. The storage and processing is directly proportional to number of features; hence selection of critical conditions is very important to increase throughput.

A statistical correlation can produce three definite spatial relations; Shared or Variance correlations and New attribute. The observations, normalized and correlated by analyser are redirected to trainer (k-means) module - placing distribution of observations into destined clusters. The normalized output produced by analyser, with correlated segments from processed sensors, is fetched in sequence by k-mean handler and deposited in array structure. Every single data point is processed to mark each index in dual dimension space; in terms of tabular representation, rows as unique index and the columns with actual data. The k-means distributes the data-set into clusters in which each observation belongs to the cluster with the nearest mean, thus resulting data space partitioning into Voronoi cells. The number of clusters denoted by 'k' can be predefined in case of supervised method or an unsupervised technique can be employed to recursively form clusters with average standard deviation of each cluster.

```
Di = Data = {1,2,2,3,4,5} Normalized and correlated set of data
Oj = Occurrence = {space of events} Time stamp or unique record index
Sn = Cluster Boundaries, Segment size to form a cluster
Raw_Data = An array of data points
Rd = Raw Data Index
Cluster_Data = Two dimensional array of data points
Cn = Cluster Index
Cnd = Cluster Data Index

For  each Di from Data-Store
      Raw_Data[Rd] = Di
            Cn = Get_Segment_Allocation, Column index of Cluster
            Cnd = Get_Row_Index of given column
            Cluster_Data[Cn][Cnd] = Rd
            Cn++, Cnd++,Rd++
End Data-Store
```

Figure 4. k-mean Trainer Algorithm

The pseudo code in Fig. 4 illustrates the trainer module, which classifies correlated annotations into collections of classes. Here in this example, a *Di* is instance of normalized and correlated data and *Oi* is an occurrence index when the particular process or network packet was recorded. The occurrence index serves as unique coordinator in proposed procedure without any significance. Important thing to note is: uniqueness which will be required to identify the original data point with-in clustered data. The *Sn* is a size of segment that a single cluster will occupy; this serves as a filter in *k-means* system to catalogue records into various clusters. A *Raw_Data* is an original data read in form of single array with *Rd* serving as its unique index. The *Cluster_Data* is two dimensional array which holds unique index of *Cnd* consisting of cluster number *(Cn)* and Cluster row index denoted by *Cnd*,

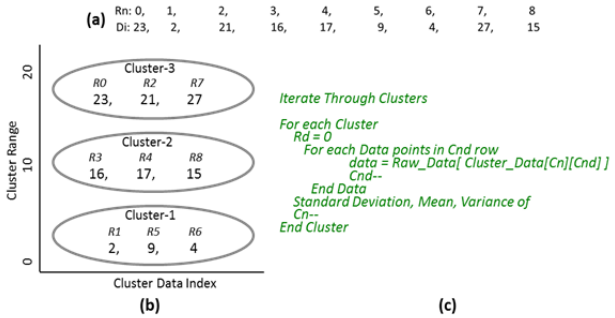

Figure 5. k-mean Classifier

Fig. 5 (a) portrays inner working of applied classifier with a sample data, *Rn* in is row index whereas *Di* is data points. The *k-means*

classifies sample data into three clusters with segment range of '*10'*. These three clusters are illustrated in Fig. 5 *(b)* with *Rn* and *Di* values. Here, *y-axis* represents the cluster range which is set to '*10'* in this example, three clusters were sufficient to distribute and classify the whole data as minimum data value is '*two*' with maximum rate of *'27'* The *x-axis* holds index of clusters' rows. In Fig. 5 *(c)* a pseudo-code is provided to iterate through these clusters.

D. Alert System – Detection and Prevention

The surveillance data correlated and produced by analyser is classified with trainer module as plotted in Fig. 6. CSA interprets these results to identify possible system attacks such as DOS, DDOS, Ghost access, ID spoofing - with rationality - of possible abnormal behaviour while examining all the events.

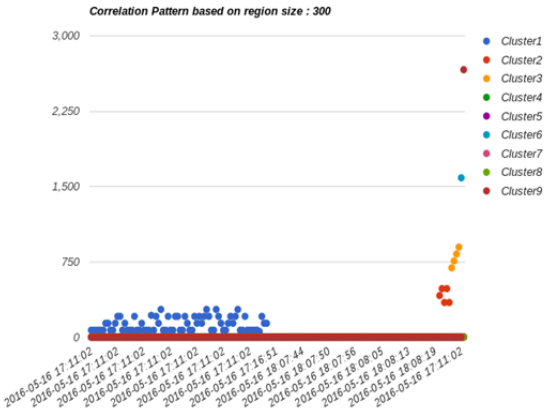

Figure 6. Output of clusters by k-mean

The formation of clusters indicates different events in the system and number of points in each cluster denotes specific occurrence of a certain event. A build-up in certain cluster with saturation of events indicates reoccurrence of same actions in point-of-time. The trainer or k-means module relies on user input that allows filtering and fine tuning clusters with segment size, number of data rows to be observed with threshold value to exclude undesired noise. A *'hitcount'* column serves as watch-dog to detect malfunctioning of normalization process. The sum of *'hitcount'* should be always equal to number of rows in original data and also equal to *'sc+vc1+vc2+vc3'*.

The input field labelled as *'Y-axis'* holds data set to be observed from correlation construction with feature selection e.g. *sc, vc1, vc2 or vc3*. The values over *y-axis* are portrayed to generate an index in two-dimension space, segment size is an input field to create size of cluster; the size means that each data point will be allocated to specific single cluster based on its value. Standard Deviation *(σ)* calculated of each cluster assists to assess appropriate Segment for each data-point. The value of σ illustrates the distribution of data points in a given cluster above and below the average dispersion. A higher numerical value of σ indicates that the data is widely spread, whereas lower sum portrays fair distribution. With network log, filters and rules of Anomaly Detection are set as follows.

1) In case of amateur DOS attack scenario, shared correlation index will be higher identifying a remote host continuously establishing a communication link, whereas *vc1* will be able to identify culprit in case of sophisticated *DOS* attack with established connections - on wait or continuous SYN (tcp request flag) requests.

2) In case of a DDOS attack, shared correlation will be very weak and transition to strong distanced variants will be an indicator of unexpected build-up in all clusters. The change in physical signature, such as MAC address, IP spoofing is typical attack vector to undermine deployed defence strategies that are based of static IDS.

As to process log, the ability of clustering provides concrete solution to identify and audit new processes in system as trainer has built process map which allows anticipating possible behaviour. Most congested cluster represents a cluster that has higher number of processes whereas least congested means a cluster with fewer number of process. Following are trigger points that can help to identify possible anomaly:

1) A process entry out of existing clusters can possibly raise alarm to identify a brand new activity that was not reported before.

2) A new process entry that matches to most congested cluster points to certain system resources being overloaded.

3) A build-up in least in cluster with fewer data-points could alarm new activities.

4. DISCUSSION

Intrusion detection and prevention is first line of defence that can be either instrumented deep in transport protocols or at applications level. The target is to efficiently revert an attack without – during and afterwards - considerable compromise in QoS. This is a big data problem, as auditing all events generated from multiple sources with immense frequency to discern malicious activity is directly proportional to storage and computation. We introduce a novel CSA framework to construct event correlations with feature selection to anticipate behaviour based on various sensors. The profiling with CSA framework summarizes the system behaviour in efficient way to identify possible anomaly.

Typical detection mechanism relies on either pattern matching or statistical analysis, whereas event correlation approach depends on combination of these two - to analyse and predict anomaly. Our proposed solution deploys both techniques with statistical analysis performed by k-mean clustering and pattern matching at feature selection.

The NetL relies on user space application to sniff raw packets based on pcap. This should be replaced with kernel modules to override incoming requests; all should be analysed before system initiates a response. For example, there is no point to raise an alarm once the attacker is able to override ssh policy or forked a phony process.

Similarly, PrcL relies on time interval to capture state of processes in given system, this could lead to undetected process that can come and go within the window of capture snapshot. Reducing the capture interval to wall-to-wall without leaving any space of any new process to be initiated by system if PrcL is running in non-pre-emptive mode can be solution. The linkage between various distributions provides definite system activity with reference points from each observation. For example, the link between network packets and in-memory processes indicates that process with pid is using network sockets.

Hadoop's file system has been used that comes with typical connection over-head cost as with any remote access. Nevertheless, it is best solution in terms of extensibility and cost. HDFS's

MapReduce or typical process forking with discrete problem solving is very useful on modern multicore CPU architectures.

The detection mechanism based on big data analytics is direct function of correlations that are constructed from various sensors. The stronger the correlation the more chances to detect new kind of anomalies. Presented system is proof of concept developed on portion of network traffic and in-memory processes, extending feature selections from network packets in NetL and processes in PrcL will not only improve detection but also provide absolute system state. NetL consuming full network traffic will complement all other sensors to fabricate true System Genetics which will be capable to record and refer every single event in the system that has occupied CPU time and consumed storage.

The time complexity of analyser algorithm is 'O(n^2)'. "Insert and update" on sorted store can be an option that can use composite index from multiple features. Finally, correlated data has no association with original data after normalization by analyser component. It means that there is no way to refer a correlated component to its original form. CSA does not require such association as correlated data is in-fact blue print of original data and has all the information embedded except time.

Last but not least, as with every anomaly based system, cyber security analytics architecture is also prone to false alarms when distance correlation is strong with authentic events. A system with global offering with only unique guests would not be able to constitute attack vector without performance degradation.

5. REFERENCES

[1] A. Patel, M. Taghavi, K. Bakhtiyari, and J. C. Júnior, "An intrusion detection and prevention system in cloud computing: A systematic review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 25–41, 2013.

[2] H. Mohamed, L. Adil, T. Saida, and M. Hicham, "A collaborative intrusion detection and Prevention System in Cloud Computing," in *2013 Africon, September 9-12*, Mauritius, 2013, pp. 1–5.

[3] N. M. Prajapati, A. Mishra, and P. Bhanodia, "Literature survey - IDS for DDoS attacks," in *Conference on IT in Business, Industry and Government (CSIBIG), March 8-9*, Sri Aurobindo Institute of Technology, Indore, INDIA, 2014, pp. 1–3.

[4] M. E. Kabir and J. Hu, "A statistical framework for intrusion detection system," in *11th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), Augus 19-21*, Xiamen, China, 2014, pp. 941–946.

[5] I. S. Thaseen and C. A. Kumar, "Intrusion detection model using fusion of PCA and optimized SVM," in *International Conference on Contemporary Computing and Informatics (IC3I), November 27-29*, Mysore, India, 2014, pp. 879–884.

[6] G. Shang-fu and Z. Chun-lan, "Intrusion detection system based on classification," in *IEEE International Conference on Intelligent Control, Automatic Detection and High-End Equipment, July 27-29*, Beijing, China, 2012, pp. 78–83.

[7] Z. Xue-qin, G. Chun-hua, and L. Jia-jun, "Intrusion Detection System Based on Feature Selection and Support Vector Machine," in *First International Conference on Communications and Networking in China, October 25-27*, Beijing, China, 2006, pp. 1–5.

[8] A. M. Lonea, D. E. Popescu, and H. Tianfield, "Detecting ddos attacks in cloud computing environment," *Int. J. Comput. Commun. Control*, vol. 8, no. 1, pp. 70–78, 2012.

[9] A. M. Lonea, D. E. Popescu, O. Prostean, and H. Tianfield, "Evaluation of Experiments on Detecting Distributed Denial of Service (DDoS) Attacks in Eucalyptus Private Cloud," in *Soft Computing Applications*, Springer, 2013, pp. 367–379.

[10] Y. Mehmood, M. A. Shibli, U. Habiba, and R. Masood, "Intrusion Detection System in Cloud Computing: Challenges and opportunities," in *2nd National Conference on Information Assurance (NCIA), December 11-12*, Rawalpindi, Pakistan, 2013, pp. 59–66.

[11] Z. Li, W. Sun, and L. Wang, "A neural network based distributed intrusion detection system on cloud platform," in *IEEE 2nd International Conference on Cloud Computing and Intelligence Systems*, Hangzhou, China, 2012, vol. 01, pp. 75–79.

[12] A. Maiti and S. Sivanesan, "Cloud controlled intrusion detection and burglary prevention stratagems in home automation systems," in *2nd Baltic Congress on Future Internet Communications, April 25-27*, Vilnius, Lithuania, 2012, pp. 182–186.

[13] U. Tupakula, V. Varadharajan, and N. Akku, "Intrusion Detection Techniques for Infrastructure as a Service Cloud," in *IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, December 12-14*, Sydney, Australia, December, 2011, pp. 744–751.

[14] S. Omar, A. Ngadi, and H. H. Jebur, "Machine learning techniques for anomaly detection: an overview," *Int. J. Comput. Appl.*, vol. 79, no. 2, pp. 33–41, 2013.

[15] P. Laskov, P. Düssel, C. Schäfer, and K. Rieck, "Learning intrusion detection: supervised or unsupervised?," in *Image Analysis and Processing - ICIAP , September 6–8*, Cagliari, Italy: Springer, 2005, pp. 50–57.

[16] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artif. Intell.*, vol. 97, no. 1, pp. 245–271, 1997.

[17] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang, "Systematic literature review of machine learning based software development effort estimation models," *Inf. Softw. Technol.*, vol. 54, no. 1, pp. 41–59, 2012.

[18] C. Puri and C. Dukatz, "Analyzing and Predicting Security Event Anomalies: Lessons Learned from a Large Enterprise Big Data Streaming Analytics Deployment," in *26th International Workshop on Database and Expert Systems Applications (DEXA), September 1-4*, Valencia, Spain, 2015, pp. 152–158.

[19] W. Xiong, H. Hu, N. Xiong, L. T. Yang, W.-C. Peng, X. Wang, and Y. Qu, "Anomaly secure detection methods by analyzing dynamic characteristics of the network traffic in cloud communications," *Inf. Sci. (Ny).*, vol. 258, pp. 403–415, Feb. 2014.

[20] I. Jeun, Y. Lee, and D. Won, "A Practical Study on Advanced Persistent Threats," in *Communications in*

Computer and Information Science, vol. 339, 2012, pp. 144–152.

[21] F. Valeur, G. Vigna, C. Kruegel, and R. A. Kemmerer, “Comprehensive approach to intrusion detection alert correlation,” *IEEE Trans. dependable Secur. Comput.*, vol. 1, no. 3, pp. 146–169, 2004.

[22] Q. Wu, D. Ferebee, Y. Lin, and D. Dasgupta, “Monitoring security events using integrated correlation-based techniques,” in *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies, April 13 - 15*, Knoxville, TN, USA, 2009, p. 47.

[23] G. Jiang and G. Cybenko, “Temporal and spatial distributed event correlation for network security,” in *American Control Conference, Proceedings of the, June 30 – Jul 2*, Boston, MA, USA, 2004, vol. 2, pp. 996–1001.

[24] M. Irfan, H. Abbas, and W. Iqbal, “Feasibility analysis for incorporating/deploying SIEM for forensics evidence collection in cloud environment,” in *Computer and Information Science (ICIS), IEEE/ACIS 14th International Conference on, June 28-July 1*, Las Vegas, NV, USA, 2015, pp. 15–21.